



Writing Effective Test Oracles

A. Hamam

Overview

- Test Oracle: Expected results encoding
- Test automation must encode testing results correctly
- Test run by hand, expected results could be explicitly written in the textual test script or left up to human tester to decide at run time
- Automated test must have that knowledge encoded as an explicit check (for e.g JUnit assert)
- Too much checking (unnecessary cost) vs. too little (not revealing failures) .

What should be checked

- By hand: human testers can observe the results
- Testers would use judgment to decide whether the software behaved correctly (according to their understanding of requirements)
- If tester could not see enough outputs to assess behavior they could add print statements
- Testers do miss failures and this is one advantage of automation
- Disadvantage of automation the correct behavior for each test must be known before the test and the results must be encoded
 - Junit encodes the oracle as a java insertion

Which tests to include in regression testing

- Automation creates a challenging question to testers
 - What should I check?
- Many form of output to check (how much should we check with an automated test)
 - Screen
 - File
 - DB
 - Messages and signals
 - Return statements, parameters , non-local variables for unit test
- Another cost vs. benefits tradeoff

Which tests to include in regression testing

- Test oracle strategy is a rule or a set of rules that specify which program states to check
- Two general characteristics
 - Precision refers to how much of the output state should be checked. Checking more of the output state increases precision.
 - Frequency refers to when and how often the output state should be checked . A low frequency means checking the output only after execution ends, checking the output state early and often would be high frequency.

Guidelines for including tests

- First: It is important to check some output
 - Not just checking crashes and exceptions as some s/w companies do (null oracle strategy).
 - The null oracle strategy is inexpensive but not efficient.

Guidelines for including tests

- Second it is important to check the right outputs.
 - Poor assertions check outputs that are not likely to be affected by the test
 - Excellent assertions check outputs that are directly affected by the test.
- Which parts of the output state are affected by the test depends on testing level
 - At unit testing level checking the return values of the methods and returned parameter values are almost always enough
 - At the system level, it is usually sufficient to check the directly visible output such as to the screen
 - Outputs to files and DB and sensors are less observable meaning assertions that checks them are difficult and expensive to write.
 - Sometimes though is necessary to check the above

Guidelines for including tests

- Third, it is not necessary to check a lot of outputs.
 - Low precision is okay
 - Checking additional inputs increase revealability very little (other than the right outputs)

Guidelines for including tests

- Fourth, it is not necessary to check the output state multiple times
 - Low frequency is Okay
 - The vast majority of failures are revealed if the final output state is checked once